

```
list1 = [num for num in range(0,50)]
```

```
print("\nList 1:")
```

```
print(list1)
```

```
list2 = [num**2 for num in range(51)]
```

```
print("\nList 2:")
```

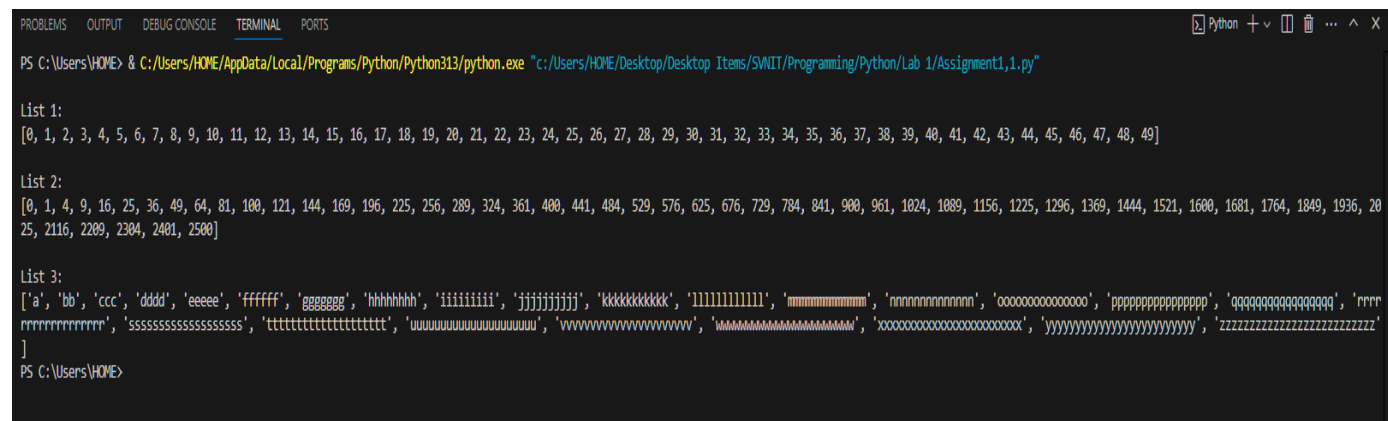
```
print(list2)
```

```
list3 = [str(chr(i))*(i - 96) for i in range(97,123)]
```

```
print("\nList 3:")
```

```
print(list3)
```

Output:



```
Python + v [icon] ... ^ X
PS C:\Users\HOME> & C:/Users/HOME/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/HOME/Desktop/Desktop Items/SWIT/Programming/Python/Lab 1/Assignment1,1.py"

List 1:
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]

List 2:
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961, 1024, 1089, 1156, 1225, 1296, 1369, 1444, 1521, 1600, 1681, 1764, 1849, 1936, 2025, 2116, 2209, 2304, 2401, 2500]

List 3:
['a', 'bb', 'ccc', 'dddd', 'eeeee', 'ffffff', 'ggggggg', 'hhhhhhh', 'iiiiiii', 'jjjjjjjj', 'kkkkkkkkk', 'llllllllll', 'mmmmmmmmmm', 'nnnnnnnnnnnn', 'oooooooooooo', 'pppppppppppppp', 'qqqqqqqqqqqqqqq', 'rrrrrrrrrrrrrrrrrr', 'ssssssssssssssss', 'tttttttttttttttt', 'uuuuuuuuuuuuuuuu', 'vvvvvvvvvvvvvvvvvv', 'wwwwwwwwwwwwwwwwww', 'xxxxxxxxxxxxxxxxxxxx', 'yyyyyyyyyyyyyyyyyyyy', 'zzzzzzzzzzzzzzzzzzzz']
PS C:\Users\HOME>
```

```
import random

mylist = []

count = 0

maxzeroes = 0

for i in range(1,101):

    random_num = random.randint(0,1)

    mylist.append(random_num)

    if (random_num == 0):

        count += 1

    else:

        if (maxzeroes < count):

            maxzeroes = count

        count = 0

print(mylist)

print(f"Longest Run of Zeroes is: {maxzeroes}")
```

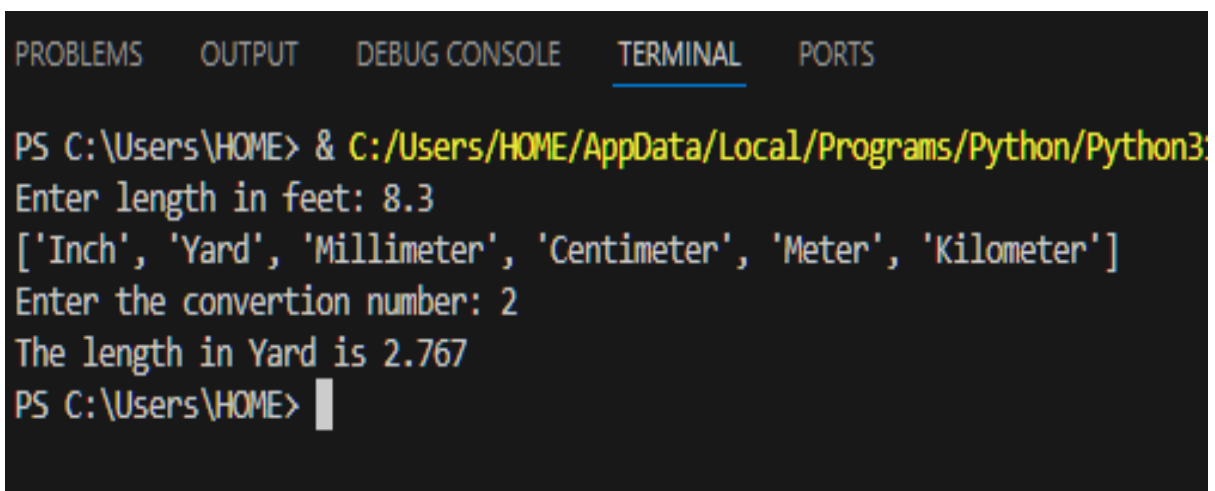
Output:

```
PS C:\Users\HOME> & C:\Users\HOME\AppData\Local\Programs\Python\Python313/python.exe "c:/Users/HOME/Desktop/Desktop Items/SWIT/Programming/Python/Lab 1/Assignment1_2.py"  
[1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0]  
Longest Run of Zeros is: 5  
PS C:\Users\HOME>
```

```
inch = 12
yard = 1/3
millimeter = 304.8
centimeter = 30.48
meter = 1000/3281
kilometer = 1/3281

convert = [inch, yard, millimeter, centimeter, meter, kilometer]
conversions = ["Inch", "Yard", "Millimeter", "Centimeter", "Meter", "Kilometer"]
feet = float(input("Enter length in feet: "))
print(conversions)
conversion = int(input("Enter the conversion number: "))
print(f"The length in {conversions[conversion-1]} is {(feet * convert[conversion-1]):.3f}")
```

Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\HOME> & C:/Users/HOME/AppData/Local/Programs/Python/Python3
Enter length in feet: 8.3
['Inch', 'Yard', 'Millimeter', 'Centimeter', 'Meter', 'Kilometer']
Enter the conversion number: 2
The length in Yard is 2.767
PS C:\Users\HOME> 
```

```

numbers = list(range(1, 10001))

equivalence_classes = {0: [], 1: [], 2: [], 3: [], 4: []}

for num in numbers:
    remainder = num % 5
    equivalence_classes[remainder].append(num)

all_numbers_in_classes = sum(equivalence_classes.values(), [])

is_valid = set(all_numbers_in_classes) == set(numbers)

print("Equivalence classes based on modulo 5:")
for remainder, class_list in equivalence_classes.items():
    print(f"Class for remainder {remainder}: {class_list[:10]}...")

if is_valid:
    print("\nThe equivalence classes are valid. The union of all equivalence classes is the original set.")
else:
    print("\nThe equivalence classes are not valid.")

```

Output:

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\HOME> & C:/Users/HOME/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/HOME/
Equivalence classes based on modulo 5:
Class for remainder 0: [5, 10, 15, 20, 25, 30, 35, 40, 45, 50]...
Class for remainder 1: [1, 6, 11, 16, 21, 26, 31, 36, 41, 46]...
Class for remainder 2: [2, 7, 12, 17, 22, 27, 32, 37, 42, 47]...
Class for remainder 3: [3, 8, 13, 18, 23, 28, 33, 38, 43, 48]...
Class for remainder 4: [4, 9, 14, 19, 24, 29, 34, 39, 44, 49]...

The equivalence classes are valid. The union of all equivalence classes is the original set.
PS C:\Users\HOME> █

```

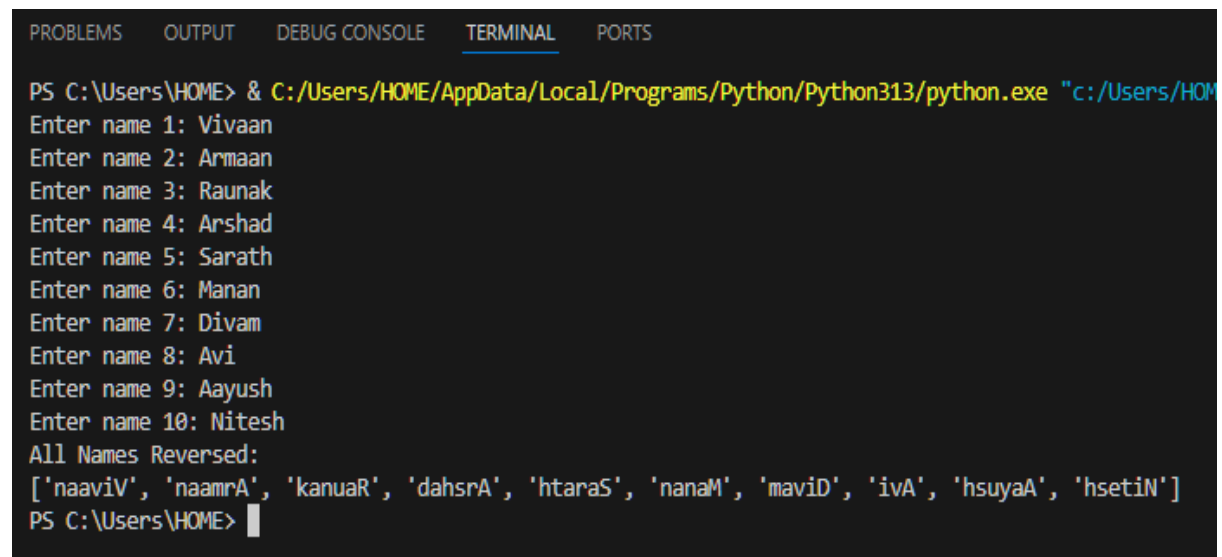
```
names = []
reversed_names = []

for i in range(1,11):
    name = str(input(f"Enter name {i}: "))
    while (len(name) > 15):
        name = str(input(f"Enter a name with less than 15 characters: "))
    names.append(name)

reversed_names = [name[::-1] for name in names]

print("All Names Reversed:")
print(reversed_names)
```

Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\HOME> & C:/Users/HOME/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/HOME/
Enter name 1: Vivaan
Enter name 2: Armaan
Enter name 3: Raunak
Enter name 4: Arshad
Enter name 5: Sarath
Enter name 6: Manan
Enter name 7: Divam
Enter name 8: Avi
Enter name 9: Aayush
Enter name 10: Nitesh
All Names Reversed:
['naaviV', 'naamrA', 'kanuaR', 'dahsrA', 'htaraS', 'nanaM', 'maviD', 'ivA', 'hsuyaA', 'hsetiN']
PS C:\Users\HOME> █
```

```

import math

def euclidean_distance(point1, point2):
    return math.sqrt((point2[0] - point1[0])**2 + (point2[1] - point1[1])**2 + (point2[2] -
point1[2])**2)

points = []

for i in range(10):
    x, y, z = map(int, input(f"Enter the coordinates for point {i + 1} (x, y, z): ").split())
    points.append((x, y, z))

nearest_neighbors = []

for i, point in enumerate(points):
    min_distance = float('inf')
    nearest_point = None

    for j, other_point in enumerate(points):
        if i != j:
            distance = euclidean_distance(point, other_point)

            if distance < min_distance:
                min_distance = distance
                nearest_point = other_point

    nearest_neighbors.append((point, nearest_point))

print("\nPoint and its nearest neighbor:")

for point, neighbor in nearest_neighbors:
    print(f"Point {point} -> Nearest Neighbor {neighbor}")

```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\Users\HOME> & C:/Users/HOME/AppData/Local/Programs/Python/Python39-64/Python.exe
Enter the coordinates for point 1 (x, y, z): 1 2 3
Enter the coordinates for point 2 (x, y, z): 4 5 6
Enter the coordinates for point 3 (x, y, z): -1 -2 -3
Enter the coordinates for point 4 (x, y, z): 7 8 9
Enter the coordinates for point 5 (x, y, z): -4 -5 -6
Enter the coordinates for point 6 (x, y, z): 10 11 12
Enter the coordinates for point 7 (x, y, z): 2 3 4
Enter the coordinates for point 8 (x, y, z): -7 -8 -9
Enter the coordinates for point 9 (x, y, z): 5 6 7
Enter the coordinates for point 10 (x, y, z): -3 -2 -1

Point and its nearest neighbor:
Point (1, 2, 3) -> Nearest Neighbor (2, 3, 4)
Point (4, 5, 6) -> Nearest Neighbor (5, 6, 7)
Point (-1, -2, -3) -> Nearest Neighbor (-3, -2, -1)
Point (7, 8, 9) -> Nearest Neighbor (5, 6, 7)
Point (-4, -5, -6) -> Nearest Neighbor (-1, -2, -3)
Point (10, 11, 12) -> Nearest Neighbor (7, 8, 9)
Point (2, 3, 4) -> Nearest Neighbor (1, 2, 3)
Point (-7, -8, -9) -> Nearest Neighbor (-4, -5, -6)
Point (5, 6, 7) -> Nearest Neighbor (4, 5, 6)
Point (-3, -2, -1) -> Nearest Neighbor (-1, -2, -3)
PS C:\Users\HOME> █
```